Logminer操作

LogMiner介绍

> Oracle LogMiner 是Oracle公司从产品8i以后提供的一个实际非常有用的分析工具，使用该工具可以轻松获得Oracle 在线/归档日志文件中的具体内容，特别是该工具可以分析出所有对于数据库操作的DML和DDL语句。该工具特别适用于调试、审计或者回退某个特定的事务。
>
> LogMiner分析工具实际上是由一组PL/SQL包和一些动态视图（Oracle8i内置包的一部分）组成，它作为Oracle数据库的一部分来发布是8i产品提供的一个完全免费的工具。但该工具和其他Oracle内建工具相比使用起来显得有些复杂，主要原因是该工具没有提供任何的图形用户界面（GUI）。

> Figure shows a sample LogMiner configuration. In this figure, the source database in Boston generates redo log files that are archived and shipped to a database in San Francisco. A LogMiner dictionary has been extracted to these redo log files. The mining database, where LogMiner will actually analyze the redo log files, is in San Francisco. The Boston database is running Oracle9i, and the San Francisco database is running Oracle Database 10g.

LogMiner作用

> 在Oracle 8i之前，Oracle没有提供任何协助数据库管理员来读取和解释重作日志文件内容的工具。系统出现问题，对于一个普通的数据管理员来讲，唯一可以作的工作就是将所有的log文件打包，然后发给Oracle公司的技术支持，然后静静地等待Oracle 公司技术支持给我们最后的答案。然而从8i以后，Oracle提供了这样一个强有力的工具–LogMiner。
>
> LogMiner 工具即可以用来分析在线，也可以用来分析离线日志文件，即可以分析本身自己数据库的重作日志文件，也可以用来分析其他数据库的重作日志文件。
>
> 总的说来，LogMiner工具的主要用途有：

> 1、跟踪数据库的变化：可以离线的跟踪数据库的变化，而不会影响在线系统的性能。
> 2、回退数据库的变化：回退特定的变化数据，减少point-in-time recovery的执行。
> 3、优化和扩容计划：可通过分析日志文件中的数据以分析数据增长模式。

LogMiner安装

运行dbmslm, dbmslmd包

> - `$ORACLE_HOME/rdbms/admin/dbmslm.sql`
> - `$ORACLE_HOME/rdbms/admin/dbmslmd.sql`
>
> 这两个脚本必须均以DBA用户身份运行。其中第一个脚本用来创建DBMS_LOGMNR包，该包用来分析日志文件。第二个脚本用来创建DBMS_LOGMNR_D包，该包用来创建数据字典文件。

类型	名称	用途
过程	Dbms_logmnr_d.build	创建一个数据字典文件
过程	Dbms_logmnr.add_logfile	在类表中增加日志文件以供分析
过程	Dbms_logmnr.start_logmnr	使用一个外部的字典文件和前面确定要分析日志文件来启动LogMiner
过程	Dbms_logmnr.end_logmnr	停止LogMiner分析
视图	V\$logmnr_dictionary	显示用来决定对象ID名称的字典文件的信息
视图	V\$logmnr_logs	在LogMiner启动时显示分析的日志列表
视图	V\$logmnr_contents	LogMiner启动后，查询重做日志的内容

创建数据字典文件

> LogMiner工具实际上是由两个新的PL/SQL内建包（(DBMSLOGMNR 和 DBMS LOGMNR_D）和四个V$动态性能视图（视图是在利用过程DBMS_LOGMNR.START_LOGMNR启动LogMiner时创建）组成。在使用LogMiner工具分析redo log文件之前，可以使用DBMS_LOGMNR_D 包将数据字典导出为一个文本文件。该字典文件是可选的，但是如果没有它，LogMiner解释出来的语句中关于数据字典中的部分（如表名、列名等）和数值都将是16进制的形式，我们是无法直接理解的。例如，下面的sql语句：
> `INSERT INTO dm_dj_swry (rydm, rymc) VALUES (00005, '张三');`
>
> LogMiner解释出来的结果将是下面这个样子：
> `insert into Object#308(col#1, col#2) values (hextoraw('c30rte567e436'), hextoraw('4a6f686e20446f65'));`
>
> 创建数据字典的目的就是让LogMiner引用涉及到内部数据字典中的部分时为他们实际的名字，而不是系统内部的16进制。数据字典文件是一个文本文件，使用包DBMS_LOGMNR_D来创建。如果我们要分析的数据库中的表有变化，影响到库的数据字典也发生变化，这时就需要重新创建该字典文件。另外一种情况是在分析另外一个数据库文件的重作日志时，也必须要重新生成一遍被分析数据库的数据字典文件。

```
#创建数据字典文件之前需要配置LogMiner文件夹：
SQL> create directory utlfile as '/home/oracle/app/oracle/diag/logmnr';

Directory created.

SQL> show parameter utl_file_dir;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
utl_file_dir                         string

SQL> alter system set utl_file_dir='/home/oracle/app/oracle/diag/logmnr' scope=spfile;

System altered.

#需要重启DB，才会s
SQL> show parameter utl_file_dir;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
utl_file_dir                         string

SQL> shutdown immediate;
```

```
Database closed.
Database dismounted.
ORACLE instance shut down.
SQL>
SQL>
SQL> startup;
ORACLE instance started.

Total System Global Area 6664212480 bytes
Fixed Size                  2239072 bytes
Variable Size            4311745952 bytes
Database Buffers         2332033024 bytes
Redo Buffers               18194432 bytes
Database mounted.
Database opened.
SQL>
SQL> show parameter utl_file_dir;

NAME                                 TYPE        VALUE
------------------------------------ ----------- ------------------------------
utl_file_dir                         string      /home/oracle/app/oracle/diag/l
                                                 ogmnr
#以DBA用户登录，创建数据字典文件，也称为外部数据字典
SQL> EXECUTE dbms_logmnr_d.build(dictionary_filename => 'dictionary.ora', dictionary_location =>'/home/oracle/app/oracle/diag/logmnr');

PL/SQL procedure successfully completed.

```

加入需分析的日志文件

> Oracle的LogMiner可以分析在线（online）和归档（offline）两种日志文件，加入分析日志文件使用dbms_logmnr.add_logfile过程，第一个文件使用dbms_logmnr.NEW参数，后面文件使用dbms_logmnr.ADDFILE参数。

```
#添加redo log日志分析
```

```
begin
    dbms_logmnr.add_logfile(logfilename=>'/home/oracle/app/oracle/oradata/mmpdb3/redo02.log', options=>dbms_logmnr.NEW);
    dbms_logmnr.add_logfile(logfilename=>'/home/oracle/app/oracle/oradata/mmpdb3/redo01.log', options=>dbms_logmnr.ADDFILE);
    dbms_logmnr.add_logfile(logfilename=>'/home/oracle/app/oracle/oradata/mmpdb3/redo03.log', options=>dbms_logmnr.ADDFILE);
end;
/

#添加归档日志分析
SQL> select name from v$archived_log where first_time > to_date('2017-1-5 00:00:00','YYYY-MM-DD HH24:MI:SS');

NAME
--------------------------------------------------------------------------------
/home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog/2017_01_05/o1_mf_1_196_d6vnqpo3_.arc
/home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog/2017_01_05/o1_mf_1_197_d6vpn2xx_.arc
/home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog/2017_01_05/o1_mf_1_198_d6vpn9fb_.arc

begin
    dbms_logmnr.add_logfile(logfilename=>'/home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog/2017_01_05/o1_mf_1_198_d6vpn9fb_.ar
end;
/
```

使用LogMiner进行日志分析

> Oracle的LogMiner分析时分为无限制条件和限制条件两种，无限制条件中分析所有加入到分析列表日志文件，限制条件根据限制条件分析指定范围日志文件。
>
> 1、无限制条件

```
SQL> execute dbms_logmnr.start_logmnr(dictfilename=>'/home/oracle/app/oracle/diag/logmnr/dictionary.ora');

PL/SQL procedure successfully completed.
```

> 2、有条件的

```
ALTER SESSION SET NLS_DATE_FORMAT = 'YYYY-MM-DD HH24:MI:SS';

EXECUTE DBMS_LOGMNR.START_LOGMNR( -
   DICTFILENAME => '/home/oracle/app/oracle/diag/logmnr/dictionary.ora', -
   STARTTIME => '2017-1-5 08:30:00', -
   ENDTIME => '2017-1-5 09:00:00');
```

观察分析结果（v$logmnr_contents)

```
SQL> select timestamp, username, table_name, SQL_REDO from v$logmnr_contents where table_name = 'TLOG';

TIMESTAMP USERNAME   TABLE_NAME      SQL_REDO
--------- ---------- --------------- ----------------------------------------------------------------
05-JAN-17 MP         TLOG            insert into "MP"."TLOG"("ID","NAME") values ('10','log10');
```

> 需要强调一点的是，视图v$logmnr_contents中的分析结果仅在我们运行过程’dbms_logmrn.start_logmnr’这个会话的生命期中存在。这是因为所有的LogMiner存储都在PGA内存中，所有其他的进程是看不到它的，同时随着进程的结束，分析结果也随之消失。
>
> 最后，使用过程SYS.DBMS_LOGMNR.END_LOGMNR终止日志分析事务，此时PGA内存区域被清除，分析结果也随之不再存在。

```
SQL> EXECUTE SYS.DBMS_LOGMNR.END_LOGMNR;

PL/SQL procedure successfully completed.

SQL> select timestamp, username, table_name, SQL_REDO from v$logmnr_contents where table_name = 'TLOG';
select timestamp, username, table_name, SQL_REDO from v$logmnr_contents where table_name = 'TLOG'
*
ERROR at line 1:
ORA-01306: dbms_logmnr.start_logmnr() must be invoked before selecting from v$logmnr_contents
```

logminer 遇到问题

> **issue1:** 查找v$logmnr_contents表没有找到insert,update 操作的记录
>
> 使用logmnr分析日志，查找v$logmnr_contents表，发现不到对表的insert ,update记录，后来才知道原来是Supplemental logging is turned off。

```
SQL> select supplemental_log_data_pk, supplemental_log_data_ui from v$database;
SUPPLEMENTAL_LOG_DATA_PK SUPPLEMENTAL_LOG_DATA_UI
------------------------ ------------------------
NO NO
对数据库作如下修改:
SQL> alter database add supplemental log data (primary key, unique index) columns;
Database altered
SQL> select supplemental_log_data_pk, supplemental_log_data_ui from v$database;
SUPPLEMENTAL_LOG_DATA_PK SUPPLEMENTAL_LOG_DATA_UI
------------------------ ------------------------
YES YES
```

实例

> 实例一：使用LogMiner读取在线日志

```
#在数据库创建LOGMINER用户，该用户需要具有DBA权限
SQL> create user logminer identified by pass;

User created.

SQL> grant connect, resource, dba to logminer;

Grant succeeded.

#以mp用户，进行测试数据准备
SQL> create table t_redo(id number, name varchar(20));
```

```
Table created.

SQL> insert into t_redo values(1, 'redo log 1');

1 row created.

SQL> insert into t_redo values(2, 'redo log 2');

1 row created.

SQL> commit;

Commit complete.

#创建数据字典文件
#数据库对象发生变化，需要重新创建数据字典文件
#以LOGMINER用户（DBA权限）登录，生成字典文件
[oracle@hzvscmdb ~]$ sqlplus logminer/pass as sysdba

SQL> execute dbms_logmnr_d.build(dictionary_filename => 'dictionary.ora', dictionary_location => '/home/oracle/app/oracle/diag/logmnr');

PL/SQL procedure successfully completed.

#确认当前处于联机状态的日志文件
SQL> select group#, sequence#, status, first_change#, first_time from v$log order by first_change#;

    GROUP#  SEQUENCE# STATUS           FIRST_CHANGE# FIRST_TIM
---------- ---------- ---------------- ------------- ---------
         2        203 INACTIVE              6691353 05-JAN-17
         3        204 INACTIVE              6691760 05-JAN-17
         1        205 CURRENT               6699917 06-JAN-17

SQL> select GROUP#, STATUS, TYPE, MEMBER from v$logfile;

    GROUP# STATUS  TYPE    MEMBER
```

```
---------- ------- ------- ----------------------------------------------------------
         1         ONLINE  /home/oracle/app/oracle/oradata/mmpdb3/redo01.log
         3         ONLINE  /home/oracle/app/oracle/oradata/mmpdb3/redo03.log
         2         ONLINE  /home/oracle/app/oracle/oradata/mmpdb3/redo02.log
         1         ONLINE  /home/oracle/app/oracle/oradata/mmpdb3/redo01-1.log

#加入解析在线日志文件
begin
    dbms_logmnr.add_logfile(logfilename=>'/home/oracle/app/oracle/oradata/mmpdb3/redo01.log', options=>dbms_logmnr.NEW);
end;
/


#启动LogMiner进行分析
execute dbms_logmnr.start_logmnr(dictfilename=>'/home/oracle/app/oracle/diag/logmnr/dictionary.ora');

#观察分析结果
SQL> select sql_redo, sql_undo, table_name, username from v$logmnr_contents where seg_name='T_REDO';

SQL_REDO                          SQL_UNDO                          TABLE_NAME USERNAME
--------------------------------- --------------------------------- ---------- ----------
create table t_redo(id number, name                                 T_REDO     MP
 varchar(20));

insert into "MP"."T_REDO"("ID","NAM delete from "MP"."T_REDO" where "ID T_REDO     MP
E") values ('1','redo log 1');      " = '1' and "NAME" = 'redo log 1' a
                                    nd ROWID = 'AAATeZAAFAAA6OTAAA';

insert into "MP"."T_REDO"("ID","NAM delete from "MP"."T_REDO" where "ID T_REDO     MP
E") values ('2','redo log 2');      " = '2' and "NAME" = 'redo log 2' a
                                    nd ROWID = 'AAATeZAAFAAA6OTAAB';
```

> 实例二：使用LogMiner读取归档日志

```
[oracle@hzvscmdb ~]$ sqlplus mp/pass

SQL> create table t_arch(id number, name varchar(20));

Table created.

SQL> insert into t_arch values(1, 'arch log 1');

1 row created.

SQL> insert into t_arch values(2, 'arch log 2');

1 row created.

SQL> commit;

Commit complete.

#从v$log视图中找出日志文件的序号
[oracle@hzvscmdb ~]$ sqlplus logminer/pass as sysdba

SQL> select GROUP#, SEQUENCE#, ARCHIVED, STATUS, FIRST_CHANGE#, FIRST_TIME, NEXT_CHANGE#, NEXT_TIME from v$log;

    GROUP#  SEQUENCE# ARC STATUS           FIRST_CHANGE# FIRST_TIM         NEXT_CHANGE# NEXT_TIME
---------- ---------- --- ---------------- ------------- --------- -------------------- ---------
         1        205 YES INACTIVE              6699917 06-JAN-17              6718064 06-JAN-17
         2        206 NO  CURRENT               6718064 06-JAN-17      281474976710655
         3        204 YES INACTIVE              6691760 05-JAN-17              6699917 06-JAN-17

SQL> select SEQUENCE#, FIRST_TIME, NEXT_CHANGE#, NEXT_TIME, STATUS, NAME from v$archived_log order by NEXT_TIME desc;

 SEQUENCE# FIRST_TIM NEXT_CHANGE# NEXT_TIME S NAME
---------- --------- ------------ --------- - ------------------------------------------------------------
       205 06-JAN-17      6718064 06-JAN-17 A /home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog
                                              /2017_01_06/o1_mf_1_205_d6y93ow0_.arc

```

```
        204 05-JAN-17      6699917 06-JAN-17 A /home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog

                                               /2017_01_06/o1_mf_1_204_d6xrnj1p_.arc

        203 05-JAN-17      6691760 05-JAN-17 D
        202 05-JAN-17      6691353 05-JAN-17 D


SQL> alter system switch logfile;

System altered.


SQL> select GROUP#, SEQUENCE#, ARCHIVED, STATUS, FIRST_CHANGE#, FIRST_TIME, NEXT_CHANGE#, NEXT_TIME from v$log;

    GROUP#  SEQUENCE# ARC STATUS           FIRST_CHANGE# FIRST_TIM         NEXT_CHANGE# NEXT_TIME
---------- ---------- --- ---------------- ------------- --------- -------------------- ---------
         1        205 YES INACTIVE               6699917 06-JAN-17              6718064 06-JAN-17
         2        206 YES ACTIVE                 6718064 06-JAN-17              6720936 06-JAN-17
         3        207 NO  CURRENT                6720936 06-JAN-17      281474976710655

SQL> select SEQUENCE#, FIRST_TIME, NEXT_CHANGE#, NEXT_TIME, STATUS, NAME from v$archived_log order by NEXT_TIME desc;

 SEQUENCE# FIRST_TIM         NEXT_CHANGE# NEXT_TIME S NAME
---------- --------- -------------------- --------- - ------------------------------------------------------------
        206 06-JAN-17              6720936 06-JAN-17 A /home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog
                                                       /2017_01_06/o1_mf_1_206_d6ydh37f_.arc

        205 06-JAN-17              6718064 06-JAN-17 A /home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog
                                                       /2017_01_06/o1_mf_1_205_d6y93ow0_.arc

        204 05-JAN-17              6699917 06-JAN-17 A /home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog
                                                       /2017_01_06/o1_mf_1_204_d6xrnj1p_.arc

        203 05-JAN-17              6691760 05-JAN-17 D
        202 05-JAN-17              6691353 05-JAN-17 D
```

```
#创建数据字典文件
#数据库对象发生变化，需要重新创建数据字典文件
#以LOGMINER用户（DBA权限）登录，生成字典文件
[oracle@hzvscmdb ~]$ sqlplus logminer/pass as sysdba

SQL> execute dbms_logmnr_d.build(dictionary_filename => 'dictionary.ora', dictionary_location => '/home/oracle/app/oracle/diag/logmnr');

PL/SQL procedure successfully completed.

#加入需分析的归档日志文件
#加入解析在线日志文件
begin
    dbms_logmnr.add_logfile(logfilename=>'/home/oracle/app/oracle/fast_recovery_area/MMPDB3/archivelog/2017_01_06/o1_mf_1_206_d6ydh37f_.a
end;
/

#启动LogMiner进行分析
execute dbms_logmnr.start_logmnr(dictfilename=>'/home/oracle/app/oracle/diag/logmnr/dictionary.ora');

#观察分析结果
SQL> select sql_redo, sql_undo, table_name, username from v$logmnr_contents where seg_name='T_ARCH';

SQL_REDO                          SQL_UNDO                          TABLE_NAME      USERNAME
--------------------------------- --------------------------------- --------------- ----------
create table t_arch(id number, name                                 T_ARCH          MP
 varchar(20));

insert into "MP"."T_ARCH"("ID","NAM delete from "MP"."T_ARCH" where "ID T_ARCH          MP
E") values ('1','arch log 1');      " = '1' and "NAME" = 'arch log 1' a
                                    nd ROWID = 'AAATeaAAFAAA6ObAAA';

insert into "MP"."T_ARCH"("ID","NAM delete from "MP"."T_ARCH" where "ID T_ARCH          MP
E") values ('2','arch log 2');      " = '2' and "NAME" = 'arch log 2' a
                                    nd ROWID = 'AAATeaAAFAAA6ObAAB';
```